**CSP BYPASS**

Content Security Policy (CSP) is used to define where scripts and other resources can be loaded or executed from. This module will walk you through ways to bypass the policy based on common mistakes made by developers.

None of the vulnerabilities are actual vulnerabilities in CSP, they are vulnerabilities in the way it has been implemented.

**Description:**Content Security Policy or CSP is a built-in browser technology which helps protect from attacks such as cross-site scripting (XSS). It lists and describes paths and sources, from which the browser can safely load resources. The resources may include images, frames, javascript and more.

**Objective:**

Bypass Content Security Policy (CSP) and execute JavaScript in the page.

**Impact:** A quick analysis reveals the following: The CSP commands unsafe-inline and unsafe-eval allow inline scripts and scripts from event attributes to execute, something that is highly damaging to the website's client-site security. Really, the only good thing about the header above is that it enforces HTTPS.

**Prevention:**

* 1. If you have a limited list of destination pages to redirect or forward to, store full URLs in the database, give them identifiers, and use the identifiers as request parameters, redirecting to the actual URL represented by an identifier. With such an approach, attackers will not be able to redirect or forward to unauthorized pages.
  2. If you cannot use a list of destination pages and/or URLs, filter untrusted URL input, preferably on the basis of a whitelist, not a blacklist. Be careful when checking for partial strings, for example, note that *http://example.com.evil.com* is a valid URL. Also, allow only HTTP and HTTPS protocols. Note that this solution is risky because errors in filtering may make certain attack vectors possible.

**LOW**

**Steps to reproduce:**

**First method**

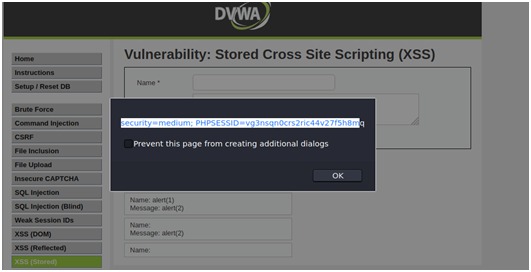
1. Configure your browser

2. Go to the dvwa page and set level of CSP bypass to the low level.

3. Go to the file upload section and upload the file.js and copy the file path from there.

4. Access the file through “../../hackable/uploads/file.js”.

5. Get pop up of alert hi.





**Second Method**

1. Configure your browser

2. Go to the dvwa page and set level of CSP bypass to the low level.

3. Create a file name file.js in which alert(“hi”); stored and copied to DVWA and html directory.

4. Now access the file in DVWA directory through “127.0.0.1/DVWA/file.js”

5. Get an pop up of alert hi.





**MEDIUM**

**Steps to reproduce:**

1. Configure your browser

2. Go to the dvwa page and set level of CSP bypass to the medium level.

3. In source code it is written that only script tag with nonce will run.

4. <script nonce="TmV2ZXIgZ29pbmcgdG8gZ2l2ZSB5b3UgdXA=">alert(1)</script>

5. Get a pop up of alert 1.

6. For user cookies “<script nonce="TmV2ZXIgZ29pbmcgdG8gZ2l2ZSB5b3UgdXA=">alert(document.cookie)</script>”



**HIGH**

**Steps to reproduce:**

1. Configure your browser and burp suite.

2. Go to the dvwa page and set level of CSP bypass to the high level.

3. Click on the solve the sum and capture the request in tool like burp suite.

4. In request change the value of callback parameter to alert(1).

5. Get a pop up of alert 1.

6. For user cookies alert(document.cookie).

